WE CLAIM:

1. A method of loading program code, data, and control information into a processing engine, said method comprising:

receiving said program code, said data, and said control information via a single input data path, said program code, said data, and said control information including identification bits;

reading an identification bit received via said input data path, said identification bit indicating one of setup data and program data;

processing setup data received via said input data path in accordance with said identification bit indicating setup data; and

processing program data received via said input data path in accordance with said identification bit indicating program data.

2. The method of claim 1 wherein said receiving said program code, said data, and said control information comprises receiving pieces of program code, data, and control information, wherein each said piece includes said identification bit and at least one instruction corresponding to said identification bit.

3. The method of claim 1 wherein said reading an identification bit further comprises reading an identification bit corresponding to setup data.

4. The method of claim 1 wherein said processing setup data comprises:

identifying a piece of storage in said processing engine, wherein said piece of storage is one of a register file, a memory, and a program counter; and

supplying a value to be stored in said piece of storage.

5. The method of claim 1 wherein said processing setup data comprises:

loading a memory address of a first instruction of a program; and

loading said program in a block of memory, wherein said loading includes storing a set of instructions in sequential order beginning with said first instruction at said memory address and incrementing said memory address for each subsequent instruction.

6. The method of claim 1 wherein said processing setup data comprises:

loading a value into a register file counter;

loading a constant at said value; and

incrementing said register file counter for loading a second constant.

7. The method of claim 1 wherein said reading an identification bit further comprises reading an identification bit corresponding to program data.

8. The method of claim 1 wherein said processing program data comprises:

fetching an instruction of a program at a memory address;

decoding said instruction;

executing said instruction; and

storing a result from said executing said instruction.

9. The method of claim 8 further comprising incrementing said memory address to fetch a second instruction.

10. The method of claim 1 further comprising propagating an input value from said single input data path through an execution pipeline to an output data path without said input value being changed.

11. The method of claim 10 further comprising generating output data identification signals to allow said input value to be interleaved with said data.

12. A method of loading setup data and program data into a computer processing engine, said method comprising:

receiving said setup data and said program data in a single input data stream, said setup data and said program data each including pieces of input data;

providing identification of each piece of input data to indicate whether said piece of input data is setup data or program data;

automatically switching said processing engine from setup mode to run mode when said identification indicates program data after a most recent prior identification had indicated setup data; and

automatically switching said processing engine from run mode to setup mode when said identification indicates setup data after a most recent prior identification had indicated program data.

13. The method of claim 12 wherein each piece of input data includes an instruction sequence for one of setup data and program data.

14. The method of claim 12 wherein said automatically switching said processing engine from setup mode to run mode further comprises processing program data by executing a stored instruction sequence.

15. The method of claim 12 wherein said automatically switching said processing engine from run mode to setup mode further comprises processing setup data by executing a pass-through instruction that causes said setup data to propagate through an execution pipeline without modification.

16. The method of claim 12 further comprising waiting in one of a setup mode or run mode until a next said piece of input data is received.

17. A data processing engine comprising:

a single input data path for receiving information that includes program data and setup data;

an execution pipeline coupled to said input data path that performs arithmetic and logic operations; and

control logic coupled to said execution pipeline that determines whether information received on said input data path is setup data or program data.

18. The data processing engine of claim 17 wherein said information comprises pieces of input data, each piece of input data including one of:

a setup identification bit and said setup data; and

a program identification bit and said program data.

19. The data processing engine of claim 17 further comprising an input data identification path coupled to said control logic that transports setup and program identification bits.

20. The data processing engine of claim 19 wherein said execution pipeline executes a pass-through instruction allowing setup data to propagate through said execution pipeline without modification when said control logic reads a setup identification bit received via said input data identification path.

21. The data processing engine of claim 19 wherein said execution pipeline processes said program data when said control logic reads a program identification bit received via said input data identification path.

22. The data processing engine of claim 17 further comprising an input data identification path coupled to said control logic that transports identification bits indicating either setup data or program data, wherein said control logic reads an identification bit received via said input data identification path and determines whether said identification bit indicates setup data or program data.

23. A computer processing engine comprising:
an execution pipeline comprising:
a first input coupled to receive setup data and program data from the same input data path,

a second input, and
an output;
a memory comprising:
a first input coupled to said output of said execution pipeline,
a second input, and
an output;
a program counter comprising:
an input, and
an output coupled to said input of said memory;
control logic comprising:
a first input coupled to said output of said execution pipeline,
a second input coupled to said output of said memory,
a third input coupled to said output of said program counter,
a first output coupled to said program counter input, and
a second output; and
a register file comprising:
a first input coupled to said output of said execution pipeline,
a second input coupled to said second output of said control logic, and
an output coupled to said second input of said execution pipeline.

24. The computer processing engine of claim 23 wherein said execution pipeline performs arithmetic and logic operations.

25. The computer processing engine of claim 23 wherein said execution pipeline processes a pass-through instruction when data from said input data

path is setup data, said setup data propagating through said execution pipeline without modification.

26. The computer processing engine of claim 23 wherein said first input of said memory receives a set of instructions when data from said input data path is setup data.

27. The computer processing engine of claim 23 wherein said second input of said memory is coupled to receive a memory address from said program counter when data from said input data path is setup data.

28. The computer processing engine of claim 23 wherein said input of said program counter is coupled to receive a memory address from said control logic.

29. The computer processing engine of claim 23 wherein said output of said program counter comprises a memory address, said memory address comprising a write memory address when data from said input data path is setup data and a read memory address when data from said input data path is program data.

30. The computer processing engine of claim 23 wherein said control logic further comprises a fourth input coupled to receive an identification bit from an input data identification data path, said identification bit indicating whether data from said input data path is setup data or program data.

31. The computer processing engine of claim 23 wherein said control logic:

sends an instruction to said execution pipeline;

sends to said register file an address to read data from; and

sends to said register file an address to write data to.

32. The computer processing engine of claim 23 wherein said first input of said register file is coupled to receive a result from an arithmetic or logic operation performed by said execution pipeline.

33. Apparatus for loading program code, data, and control information into a processing engine, said apparatus comprising:

means for receiving said program code, said data, and said control information via a single input data path, said program code, said data, and said control information including identification bits;

means for reading an identification bit received via said input data path, said identification bit indicating one of setup data and program data;

means for processing setup data received via said input data path in accordance with said identification bit indicating setup data; and

means for processing program data received via said input data path in accordance with said identification bit indicating program data.

34. Apparatus for loading setup data and program data into a computer processing engine, said apparatus comprising:

means for receiving said setup data and said program data in a single input data stream, said setup data and said program data each including pieces of input data;

means for providing identification of each piece of input data to indicate whether said piece of input data is setup data or program data;

means for automatically switching said processing engine from setup mode to run mode when said identification indicates program data; and

means for automatically switching said processing engine from run mode to setup mode when said identification indicates setup data.

35. A data processing engine comprising:

a single input data path means for receiving information that includes program data and setup data;

execution pipeline means coupled to said input data path for performing arithmetic and logic operations; and

control logic means coupled to said execution pipeline for determining whether information received on said input data path is setup data or program data.

36. A computer processing engine comprising:

execution pipeline means for performing arithmetic and logic operations, said pipeline means comprising:

a first input coupled to receive setup data and program data from the same input data path,

a second input, and

an output;

memory means comprising:

a first input coupled to said output of said execution pipeline,

a second input, and

an output;

program counter means comprising:

an input, and

an output coupled to said input of said memory;

control logic means for determining whether information received on said input data path is setup data or program data, said control logic means comprising:

a first input coupled to said output of said execution pipeline,

a second input coupled to said output of said memory,

a third input coupled to said output of said program counter,

a first output coupled to said program counter input, and

a second output; and

register file means for providing input data to said execution pipeline and for storing output data from said execution pipeline, said register file means comprising:

a first input coupled to said output of said execution pipeline,

a second input coupled to said second output of said control logic, and

an output coupled to said second input of said execution pipeline.